```
EEEEEEEEEEEEEEE  RRRRRRRRRRRR     FFFFFFFFFFFFFFF
EEEEEEEEEEEEEEE  RRRRRRRRRRRR     FFFFFFFFFFFFFFF
EEEEEEEEEEEEEEE  RRRRRRRRRRRR     FFFFFFFFFFFFFFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEEEEEEEEEEE     RRRRRRRRRRRR     FFFFFFFFFFFF
EEEEEEEEEEEE     RRRRRRRRRRRR     FFFFFFFFFFFF
EEEEEEEEEEEE     RRRRRRRRRRRR     FFFFFFFFFFFF
EEE              RRR   RRR        FFF
EEE              RRR   RRR        FFF
EEE              RRR   RRR        FFF
EEE              RRR      RRR     FFF
EEE              RRR      RRR     FFF
EEE              RRR      RRR     FFF
EEEEEEEEEEEEEEE  RRR         RRR  FFF
EEEEEEEEEEEEEEE  RRR         RRR  FFF
EEEEEEEEEEEEEEE  RRR         RRR  FFF
```

```
**FILE**ID**IMAGELOAD                                            K  9

            IIIIII    MM      MM    AAAAAA      GGGGGGGG  EEEEEEEEEE  LL            OOOOOO      AAAAAA    DDDDDDDD
            IIIIII    MM      MM    AAAAAA      GGGGGGGG  EEEEEEEEEE  LL            OOOOOO      AAAAAA    DDDDDDDD
              II      MMMM  MMMM  AA      AA  GG          EE          LL          OO      OO  AA      AA  DD      DD
              II      MMMM  MMMM  AA      AA  GG          EE          LL          OO      OO  AA      AA  DD      DD
              II      MM  MM  MM  AA      AA  GG          EE          LL          OO      OO  AA      AA  DD      DD
              II      MM  MM  MM  AA      AA  GG          EE          LL          OO      OO  AA      AA  DD      DD
              II      MM      MM  AA      AA  GG          EEEEEEEE    LL          OO      OO  AA      AA  DD      DD
              II      MM      MM  AA      AA  GG          EEEEEEEE    LL          OO      OO  AA      AA  DD      DD
              II      MM      MM  AAAAAAAAAA  GG  GGGGGG  EE          LL          OO      OO  AAAAAAAAAA  DD      DD
              II      MM      MM  AAAAAAAAAA  GG  GGGGGG  EE          LL          OO      OO  AAAAAAAAAA  DD      DD
              II      MM      MM  AA      AA  GG      GG  EE          LL          OO      OO  AA      AA  DD      DD      ....
              II      MM      MM  AA      AA  GG      GG  EE          LL          OO      OO  AA      AA  DD      DD      ....
            IIIIII    MM      MM  AA      AA    GGGGGG    EEEEEEEEEE  LLLLLLLLLL    OOOOOO    AA      AA  DDDDDDDD        ....
            IIIIII    MM      MM  AA      AA    GGGGGG    EEEEEEEEEE  LLLLLLLLLL    OOOOOO    AA      AA  DDDDDDDD        ....

          LL            IIIIII      SSSSSSSS
          LL            IIIIII      SSSSSSSS
          LL              II      SS
          LL              II      SS
          LL              II      SS
          LL              II      SS
          LL              II        SSSSSS
          LL              II        SSSSSS
          LL              II              SS
          LL              II              SS
          LL              II              SS
          LL              II              SS
          LLLLLLLLLL    IIIIII    SSSSSSSS
          LLLLLLLLLL    IIIIII    SSSSSSSS
```

IMA
VO

```
0001  0 MODULE IMAGE_LOADER (%TITLE 'Errorlog Image Loader'
0002  0             IDENT = 'V04-000' ) =
0003  1 BEGIN
0004  1
0005  1 !****************************************************************************
0006  1 !*                                                                          *
0007  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0008  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0009  1 !*  ALL RIGHTS RESERVED.                                                    *
0010  1 !*                                                                          *
0011  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0012  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0013  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0014  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0015  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0016  1 !*  TRANSFERRED.                                                            *
0017  1 !*                                                                          *
0018  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0019  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0020  1 !*  CORPORATION.                                                            *
0021  1 !*                                                                          *
0022  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0023  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0024  1 !*                                                                          *
0025  1 !*                                                                          *
0026  1 !****************************************************************************
0027  1
0028  1 !++
0029  1 ! FACILITY:  ERF, Errorlog Report Formatter
0030  1 !
0031  1 ! ABSTRACT:
0032  1 !       This routine determins if the requested image needs to be
0033  1 !       loaded. It will load an image and initialize the transfer
0034  1 !       vectors.
0035  1 !
0036  1 !
0037  1 ! ENVIRONMENT:
0038  1 !
0039  1 !       VAX/VMS operating system. unprivileged user mode,
0040  1 !
0041  1 ! AUTHOR:  Elliott A. Drayton                   Creation date: 23-Mar-1983
0042  1 !
0043  1 ! Modified by:
0044  1 !
0045  1 !       V03-002 EAD0181         Elliott A. Drayton              27-Jun-1984
0046  1 !               Added support for workstations.
0047  1 !
0048  1 !       V03-001 EAD0161         Elliott A. Drayton              20-Apr-1984
0049  1 !               Change bind of image name in map_image.
0050  1 !
0051  1 !--
```

```
;     53        0052   1 REQUIRE 'SRC$:ERFDEF.REQ';
;     54        0338   1
;     55        0339   1 FORWARD ROUTINE
;     56        0340   1          Get_image,                            ! Gets loaded image vector information
;     57        0341   1          Image_loader,                         ! Determines which image to load
;     58        0342   1          Map_image;                            ! Translates image name and loads it.
;     59        0343   1
;     60        0344   1 EXTERNAL ROUTINE
;     61        0345   1          Get_code,                             !
;     62        0346   1          Exec_image,                           ! Call loaded image with correct params.
;     63        0347   1          Lib$map_image;
;     64        0348   1
;     65        0349   1
;     66        0350   1 GLOBAL
;     67        0351   1   Bus_image:                    REF BLOCKVECTOR[,2] FIELD (desc_fields),
;     68        0352   1   Bus_version:                  REF VECTOR[,WORD],
;     69        0353   1   Bus_xfer_addr:                REF VECTOR[,LONG],        ! Address of bus xfer address table
;     70        0354   1   Disk_image:                   REF BLOCKVECTOR[,2] FIELD (desc_fields),
;     71        0355   1   Disk_version:                 REF VECTOR[,WORD],        ! Address of version number of device dependent code
;     72        0356   1   Disk_xfer_addr:               REF VECTOR[,LONG],        ! Address of disk xfer address table
;     73        0357   1   Lp_image:                     REF BLOCKVECTOR[,2] FIELD (desc_fields),
;     74        0358   1   Lp_version:                   REF VECTOR[,WORD],
;     75        0359   1   Lp_xfer_addr:                 REF VECTOR[,LONG],        ! Address of lp xfer address table
;     76        0360   1   Max_misc_type:                BYTE,
;     77        0361   1   Max_lp_type:                  BYTE,
;     78        0362   1   Packet_processor_xfer_addr:   REF VECTOR[,LONG],        ! Address of realtime xfer address table
;     79        0363   1   Packet_processor_image:       REF BLOCKVECTOR[,2] FIELD (desc_fields),
;     80        0364   1   Packet_processor_version:     REF VECTOR[,WORD],
;     81        0365   1   Realtime_image:               REF BLOCKVECTOR[,2] FIELD (desc_fields),
;     82        0366   1   Realtime_version:             REF VECTOR[,WORD],
;     83        0367   1   Realtime_xfer_addr:           REF VECTOR[,LONG],        ! Address of realtime xfer address table
;     84        0368   1   Scom_image:                   REF BLOCKVECTOR[,2] FIELD (desc_fields),
;     85        0369   1   Scom_version:                 REF VECTOR[,WORD],
;     86        0370   1   Scom_xfer_addr:               REF VECTOR[,LONG],        ! Address of scom xfer address table
;     87        0371   1   Tape_image:                   REF BLOCKVECTOR[,2] FIELD (desc_fields),
;     88        0372   1   Tape_version:                 REF VECTOR[,WORD],
;     89        0373   1   Tape_xfer_addr:               REF VECTOR[,LONG],        ! Address of tape xfer address table
;     90        0374   1   Translate_entry_table:        REF VECTOR [,WORD],
;     91        0375   1   Workstation_image:            REF BLOCKVECTOR[,2] FIELD (desc_fields),
;     92        0376   1   Workstation_version:          REF VECTOR[,WORD],
;     93        0377   1   Workstation_xfer_addr:        REF VECTOR[,LONG],        ! Address of workstation  xfer address table
;     94        0378   1   Worst_error:                      $BBLOCK [LONG] INITIAL (1);
;     95        0379   1
;     96        0380   1
;     97        0381   1 LITERAL
;     98        0382   1   Word_size = 2,                 ! Number of bytes in a word
;     99        0383   1   Longword = 4,                  ! Number of bytes in a longword
;    100        0384   1   Descriptor_length = 8;         ! Number of bytes in a string descriptor
;    101        0385   1
;    102        0386   1
;    103        0387   1 EXTERNAL LITERAL
;    104        0388   1          Erf_loaderr,
;    105        0389   1          Erf_unkentry,
;    106        0390   1          Erf_unkclass;
;    107        0391   1
;    108        0392   1 SD ('Lnm$file_dev');
;  110         0393   1 Global routine MAP_IMAGE ( image_name, transfer_address) =
;  111         0394   2 Begin
;  112         0395   2
;  113         0396   2 !++
;  114         0397   2 ! Functional description
;  115         0398   2 !
;  116         0399   2 !       This is the control routine for the ERF facility.
;  117         0400   2 !       Any errors encountered will be passed back to this routine.
;  118         0401   2 !
;  119         0402   2 ! Calling sequence
;  120         0403   2 !
;  121         0404   2 !       ERF_CONTROL ()
;  122         0405   2 !
;  123         0406   2 ! Input parameters
;  124         0407   2 !
;  125         0408   2 !
;  126         0409   2 ! Output parameters
;  127         0410   2 !
;  128         0411   2 !       None
;  129         0412   2 !
;  130         0413   2 ! Routine value
;  131         0414   2 !
;  132         0415   2 !       Worst error is returned.
;  133         0416   2 !
;  134         0417   2 !----
;  135         0418   2
;  136         0419   2 Bind
;  137         0420   2    Image_name_addr = .image_name : $BBLOCK[dsc$k_d_bln],
;  138         0421   2    Addr = .transfer_address : LONG;
;  139         0422   2
;  140         0423   2 Literal
;  141         0424   2    Buffer_size = 80;
;  142         0425   2
;  143         0426   2 Local
;  144         0427   2    Buffer:               $BBLOCK [buffer_size],
;  145         0428   2    Desc:                 $BBLOCK [dsc$k_d_bln],
;  146         0429   2    Filename_desc:        VECTOR [2,LONG] INITIAL (buffer_size,buffer),
;  147         0430   2    Filscalst:            $itmlst_decl (items = 1),
;  148         0431   2    Status,
;  149         0432   2    Trnlnmlst:            $itmlst_decl (items = 1);
;  150         0433   2
;  151          0434  2
;  152          0435  2 !
;  153          0436  2 ! Init DESC in case there is no translation.
;  154          0437  2 !
;  155          0438  2 Desc[dsc$w_length]  = .image_name_addr[dsc$w_length];
;  156          0439  2 Desc[dsc$a_pointer] = .image_name_addr[dsc$a_pointer];
;  157          0440  2
;  158          0441  2 !
;  159          0442  2 ! Initialize a item list for a call to file scan.
;  160          0443  2 !
;  161          0444  2 $Itmlst_init ( itmlst = filscalst,(itmcod = fscn$_name,bufadr = 0,bufsiz = 0) );
;  162          0445  2
;  163          0446  2 Status = $FILESCAN (srcstr = image_name_addr, valuelst = filscalst) ;
;  164          0447  2 If NOT .status then signal_stop (.status) ;
;  165          0448  2
;  166          0449  2
;  167          0450  2 !
;  168          0451  2 ! Use the descriptor from file scan in the translate to get
;  169          0452  2 ! the real image name in filename_desc.
;  170          0453  2 !
;  171        P 0454  2 $Itmlst_init ( itmlst = trnlnmlst, (itmcod = lnm$_string,
;  172        P 0455  2                                       bufsiz = .filename_desc[0],
;  173        P 0456  2                                       bufadr = .filename_desc[1],
;  174          0457  2                                       retlen = filename_desc[0]));
;  175          0458  2
;  176        P 0459  2 If $TRNLNM ( attr = [illegible]ef(lnm$m_case_blind),
;  177        P 0460  2                 tabnam = lnm$file_dev_desc,
;  178        P 0461  2                 lognam = filscalst,
;  179          0462  3                 itmlst = trnlnmlst)
;  180          0463  2 then
;  181          0464  3    ( desc[dsc$w_length] = .filename_desc[0] ;
;  182          0465  2      desc[dsc$a_pointer] = .filename_desc[1]) ;
;  183          0466  2
;  184          0467  2 !
;  185          0468  2 ! DESC now has the original image name or its translation.
;  186          0469  2 !
;  187          0470  2 Status = LIB$MAP_IMAGE ( desc, desc, addr );
;  188          0471  2 If NOT .status then (signal (erf_loaderr, 1, desc, .status); addr = 0 );
;  189          0472  2
;  190          0473  2 Return .status;
;  191          0474  1 End;


                                                          .TITLE  IMAGE_LOADER Errorlog Image Loader
                                                          .IDENT  \V04-000\

                                                          .PSECT  $PLIT,NOWRT,NOEXE,  PIC,2

        76 65 64 5F 65 6C 69 66 24 6D 6E 4C  00000 P.AAB: .ASCII  \Lnm$file_dev\                      ;
                                  0000000C  0000C P.AAA:  .LONG   12                                  ;
                                  00000000' 00010         .ADDRESS P.AAB                              ;

                                                          .PSECT  $GLOBAL$,NOEXE,  PIC,2

                                            00000 BUS_IMAGE::
                                                          .BLKB   4
                                            00004 BUS_VERSION::
```

```
                        .BLKB   4
          00008 BUS_XFER_ADDR::
                        .BLKB   4
          0000C DISK_IMAGE::
                        .BLKB   4
          00010 DISK_VERSION::
                        .BLKB   4
          00014 DISK_XFER_ADDR::
                        .BLKB   4
          00018 LP_IMAGE::
                        .BLKB   4
          0001C LP_VERSION::
                        .BLKB   4
          00020 LP_XFER_ADDR::
                        .BLKB   4
          00024 MAX_MISC_TYPE::
                        .BLKB   1
          00025 MAX_LP_TYPE::
                        .BLKB   1
          00026         .BLKB   2
          00028 PACKET_PROCESSOR_XFER_ADDR::
                        .BLKB   4
          0002C PACKET_PROCESSOR_IMAGE::
                        .BLKB   4
          00030 PACKET_PROCESSOR_VERSION::
                        .BLKB   4
          00034 REALTIME_IMAGE::
                        .BLKB   4
          00038 REALTIME_VERSION::
                        .BLKB   4
          0003C REALTIME_XFER_ADDR::
                        .BLKB   4
          00040 SCOM_IMAGE::
                        .BLKB   4
          00044 SCOM_VERSION::
                        .BLKB   4
          00048 SCOM_XFER_ADDR::
                        .BLKB   4
          0004C TAPE_IMAGE::
                        .BLKB   4
          00050 TAPE_VERSION::
                        .BLKB   4
          00054 TAPE_XFER_ADDR::
                        .BLKB   4
          00058 TRANSLATE_ENTRY_TABLE::
                        .BLKB   4
          0005C WORKSTATION_IMAGE::
                        .BLKB   4
          00060 WORKSTATION_VERSION::
                        .BLKB   4
          00064 WORKSTATION_XFER_ADDR::
                        .BLKB   4
00000001  00068 WORST_ERROR::
                        .LONG   1                                         ;

                LNM$FILE_DEV_DESC== P.AAA
                        .EXTRN  GET_CODE, EXEC_IMAGE
```

```
                                               .EXTRN  LIB$MAP_IMAGE, ERF_LOADERR
                                               .EXTRN  ERF_UNKENTRY, ERF_UNKCLASS
                                               .EXTRN  SYS$FILESCAN, SYS$TRNLNM

                                               .PSECT  $CODE,NOWRT,  PIC,2

                           0004 00000          .ENTRY  MAP_IMAGE, Save R2                        ; 0393
           5E      FF7C  CE  9E 00002          MOVAB   -132(SP), SP                              ;
           51        04  AC  D0 00007          MOVL    IMAGE_NAME, R1                            ; 0420
       24  AE        50  8F  9A 0000B          MOVZBL  #80, FILENAME_DESC                        ; 0421
       28  AE        34  AE  9E 00010          MOVAB   BUFFER, FILENAME_DESC+4                   ;
       2C  AE            61  B0 00015          MOVW    (R1), DESC                                ; 0438
       30  AE        04  A1  D0 00019          MOVL    4(R1), DESC+4                             ; 0439
           50        14  AE  9E 0001E          MOVAB   FILSCALST, $$ITMBLKPTR                    ; 0444
           80  00060000  8F  D0 00022          MOVL    #393216, ($$ITMBLKPTR)+                   ;
                         80  7C 00029          CLRQ    ($$ITMBLKPTR)+                            ;
                         80  D4 0002B          CLRL    ($$ITMBLKPTR)+                            ;
                         7E  D4 0002D          CLRL    -(SP)                                     ; 0446
                     18  AE  9F 0002F          PUSHAB  FILSCALST                                 ;
                         51  DD 00032          PUSHL   R1                                        ;
00000000G  00            03  FB 00034          CALLS   #3, SYS$FILESCAN                          ;
           52            50  D0 0003B          MOVL    R0, STATUS                                ;
           09            52  E8 0003E          BLBS    STATUS, 1$                                ; 0447
                         52  DD 00041          PUSHL   STATUS                                    ;
00000000G  00            01  FB 00043          CALLS   #1, LIB$STOP                              ;
           50        04  AE  9E 0004A 1$:      MOVAB   TRNLNMLST, $$ITMBLKPTR                    ; 0457
           80        24  AE  B0 0004E          MOVW    FILENAME_DESC, ($$ITMBLKPTR)+             ;
           80            02  B0 00052          MOVW    #2, ($$ITMBLKPTR)+                        ;
           80        28  AE  D0 00055          MOVL    FILENAME_DESC+4, ($$ITMBLKPTR)+           ;
           80        24  AE  9E 00059          MOVAB   FILENAME_DESC, ($$ITMBLKPTR)+             ;
                         80  D4 0005D          CLRL    ($$ITMBLKPTR)+                            ;
                     04  AE  9F 0005F          PUSHAB  TRNLNMLST                                 ; 0462
                         7E  D4 00062          CLRL    -(SP)                                     ;
                     1C  AE  9F 00064          PUSHAB  FILSCALST                                 ;
              00000000'  00  9F 00067          PUSHAB  LNM$FILE_DEV_DESC                         ;
       10  AE  02000000  8F  D0 0006D          MOVL    #33554432, 16(SP)                         ;
                     10  AE  9F 00075          PUSHAB  16(SP)                                    ;
00000000G  00            05  FB 00078          CALLS   #5, SYS$TRNLNM                            ;
           0A            50  E9 0007F          BLBC    R0, 2$                                    ;
       2C  AE        24  AE  B0 00082          MOVW    FILENAME_DESC, DESC                       ; 0464
       30  AE        28  AE  D0 00087          MOVL    FILENAME_DESC+4, DESC+4                   ; 0465
                     08  AC  DD 0008C 2$:      PUSHL   TRANSFER_ADDRESS                          ; 0470
                     30  AE  9F 0008F          PUSHAB  DESC                                      ;
                     34  AE  9F 00092          PUSHAB  DESC                                      ;
00000000G  00            03  FB 00095          CALLS   #3, LIB$MAP_IMAGE                         ;
           52            50  D0 0009C          MOVL    R0, STATUS                                ;
           17            52  E8 0009F          BLBS    STATUS, 3$                                ; 0471
                         52  DD 000A2          PUSHL   STATUS                                    ;
                     30  AE  9F 000A4          PUSHAB  DESC                                      ;
                         01  DD 000A7          PUSHL   #1                                        ;
              00000000G  8F  DD 000A9          PUSHL   #ERF_LOADERR                              ;
00000000G  00            04  FB 000AF          CALLS   #4, LIB$SIGNAL                            ;
                     08  BC  D4 000B6          CLRL    @TRANSFER_ADDRESS                         ;
           50            52  D0 000B9 3$:      MOVL    STATUS, R0                                ; 0473
                             04 000BC          RET                                               ; 0474

; Routine Size:  189 bytes,    Routine Base:  $CODE + 0000
```

```
1 Global routine IMAGE_LOADER ( Device_type, Device_class, Transfer_addr ) =
2 Begin
2 !++
2 ! Functional description
2 !
2 !       This routine checks to see if the loadable image needed
2 !       to process the error packet is available. If it is not then
2 !       it calls get_image to load it. If successful then it call the
2 !       image to obtain information as to what error packets the
2 !       loaded image supports and fills in table entries which will
2 !       allow subsequent access to the modules in the image.
2 !
2 ! Calling sequence
2 !
2 !       Image_loader (Type, Class, Transfer_addr)
2 !
2 ! Input parameters
2 !
2 !
2 !
2 ! Output parameters
2 !
2 !       None
2 !
2 ! Routine value
2 !
2 !       Worst error is returned.
2 !
2 !----
2
2
2 Bind
2    Xfer_addr = .Transfer_addr,
2    Type = .Device_type : word,
2    Class = .Device_class : word;
2
2 Selectoneu .class of set
2
2 [DC$_ZERO_CLASS]:
2         ! If Device_class is zero then Device_type is really entry type. Entry types
2         ! are not contigious so they should be mapped to a valid subscript. Thus
2         ! making device_type contigious.
2
3         Begin
3           Incr I from 1 to .max_misc_type do
4             Begin
4             If .Translate_entry_table [.I] EQL .Type then
5               Begin
5               If .Packet_processor_xfer_addr [.I] EQL 0 then
5                   Get_image ( Packet_processor_image[.I, desc_one]);
5               Xfer_addr = .Packet_processor_xfer_addr [.I];
5               .Device_class = -1;
5               Exitloop;
4               End;
3             End;
3
```

```
;  250          0532  3                 If ..device_class LSS 0 then
;  251          0533  3                    .device_class = 0
;  252          0534  3                 Else
;  253          0535  4                    Begin
;  254          0536  4                    Signal (erf_unkentry, 1, .type);
;  255          0537  4                    .device_class = 0
;  256          0538  3                    End;
;  257          0539  2               End;
;  258          0540  2
;  259          0541  2
;  260          0542  2   [DC$_DISK]:
;  261          0543  3               Begin
;  262          0544  3               If .Disk_xfer_addr [.type] EQL 0 then
;  263          0545  3                  Get_image ( Disk_image[.type, desc_one] );
;  264          0546  3               Xfer_addr = .Disk_xfer_addr [.type];
;  265          0547  2               End;
;  266          0548  2
;  267          0549  2
;  268          0550  2
;  269          0551  2   [DC$_TAPE]:
;  270          0552  3               Begin
;  271          0553  3               If .Tape_xfer_addr [.type] EQL 0 then
;  272          0554  3                  Get_image ( Tape_image[.type, desc_one] );
;  273          0555  3               Xfer_addr = .Tape_xfer_addr [.type];
;  274          0556  2               End;
;  275          0557  2
;  276          0558  2
;  277          0559  2
;  278          0560  2   [DC$_SCOM]:
;  279          0561  3               Begin
;  280          0562  3               If .Scom_xfer_addr [.type] EQL 0 then
;  281          0563  3                  Get_image ( Scom_image[.type, desc_one] );
;  282          0564  3               Xfer_addr = .Scom_xfer_addr [.type];
;  283          0565  2               End;
;  284          0566  2
;  285          0567  2
;  286          0568  2
;  287          0569  2   [DC$_BUS]:
;  288          0570  3               Begin
;  289          0571  3               If .Bus_xfer_addr [.type] EQL 0 then
;  290          0572  3                  Get_image ( Bus_image[.type, desc_one] );
;  291          0573  3               Xfer_addr = .Bus_xfer_addr [.type];
;  292          0574  2               End;
;  293          0575  2
;  294          0576  2
;  295          0577  2   [DC$_REALTIME]:
;  296          0578  3               Begin
;  297          0579  3               If .Realtime_xfer_addr [.type] EQL 0 then
;  298          0580  3                  GET_IMAGE ( Realtime_image[.type, desc_one] );
;  299          0581  3               Xfer_addr = .Realtime_xfer_addr [.type];
;  300          0582  2               End;
;  301          0583  2
;  302          0584  2   [DC$_WORKSTATION]:
;  303          0585  3               Begin
;  304          0586  3               If .Workstation_xfer_addr [.type] EQL 0 then
;  305          0587  3                  GET_IMAGE ( Workstation_image[.type, desc_one] );
;  306          0588  3               Xfer_addr = .Workstation_xfer_addr [.type];
;  307              0589  2         End;
;  308              0590  2
;  309              0591  2 [OTHERWISE]:
;  310              0592  2 !
;  311              0593  2 !   If we get here then we have a unknown device class.
;  312              0594  2 !
;  313              0595  3         Begin
;  314              0596  3         Signal (erf_unkclass, 1, .class);
;  315              0597  2         End;
;  316              0598  2 TES;
;  317              0599  2 Return true;
;  318              0600  1 End;



                                 03FC 00000        .ENTRY  IMAGE_LOADER, Save R2,R3,R4,R5,R6,R7,R8,R9       ; 0475
                59 00000000G  00   9E 00002        MOVAB   LIB$SIGNAL, R9                                   ;
                58 00000000V  00   9E 00009        MOVAB   GET_IMAGE, R8                                    ;
                57 00000000'  00   9E 00010        MOVAB   PACKET_PROCESSOR_XFER_ADDR, R7                   ;
                55         0C  AC  D0 00017        MOVL    TRANSFER_ADDR, R5                                ; 0507
                54         04  AC  D0 0001B        MOVL    DEVICE_TYPE, R4                                  ; 0508
                53         08  AC  D0 0001F        MOVL    DEVICE_CLASS, R3                                 ; 0509
                               63  B5 00023        TSTW    (R3)                                             ; 0514
                               4B  12 00025        BNEQ    6$                                               ;
                56         FC  A7  9A 00027        MOVZBL  MAX_MISC_TYPE, R6                                ; 0520
                               52  D4 0002B        CLRL    I                                                ;
                               28  11 0002D        BRB     3$                                               ;
                50         30  A7  D0 0002F 1$:    MOVL    TRANSLATE_ENTRY_TABLE, R0                        ; 0522
                64           6042  B1 00033        CMPW    (R0)[I], (R4)                                    ;
                               1E  12 00037        BNEQ    3$                                               ;
                50             67  D0 00039        MOVL    PACKET_PROCESSOR_XFER_ADDR, R0                   ; 0524
                             6042  D5 0003C        TSTL    (R0)[I]                                          ;
                               0A  12 0003F        BNEQ    2$                                               ;
                50         04  A7  D0 00041        MOVL    PACKET_PROCESSOR_IMAGE, R0                       ; 0525
                             6042  7F 00045        PUSHAQ  (R0)[I]                                          ;
                68             01  FB 00048        CALLS   #1, GET_IMAGE                                    ;
                50             67  D0 0004B 2$:    MOVL    PACKET_PROCESSOR_XFER_ADDR, R0                   ; 0526
                65           6042  D0 0004E        MOVL    (R0)[I], (R5)                                    ;
                63             01  CE 00052        MNEGL   #1, (R3)                                         ; 0527
                               04  11 00055        BRB     4$                                               ; 0523
        D4      52             56  F3 00057 3$:    AOBLEQ  R6, I, 1$                                        ; 0520
                               63  D5 0005B 4$:    TSTL    (R3)                                             ; 0532
                               0E  19 0005D        BLSS    5$                                               ;
                7E             64  3C 0005F        MOVZWL  (R4), -(SP)                                      ; 0536
                               01  DD 00062        PUSHL   #1                                               ;
                   00000000G   8F  DD 00064        PUSHL   #ERF_UNKENTRY                                    ;
                69             03  FB 0006A        CALLS   #3, LIB$SIGNAL                                   ;
                               63  D4 0006D 5$:    CLRL    (R3)                                             ; 0537
                             00DE  31 0006F        BRW     22$                                              ; 0512
                01             63  B1 00072 6$:    CMPW    (R3), #1                                         ; 0542
                               1C  12 00075        BNEQ    8$                                               ;
                50         EC  A7  D0 00077        MOVL    DISK_XFER_ADDR, R0                               ; 0544
                52             64  3C 0007B        MOVZWL  (R4), R2                                         ;
                             6042  D5 0007E        TSTL    (R0)[R2]                                         ;
                               0A  12 00081        BNEQ    7$                                               ;
```

```
                    50      E4    A7  D0 00083            MOVL     DISK_IMAGE, R0                              ; 0545
                                6042  7F 00087            PUSHAQ   (R0)[R2]                                    ;
                    68            01  FB 0008A            CALLS    #1, GET_IMAGE                               ;
                    50      EC    A7  D0 0008D  7$:       MOVL     DISK_XFER_ADDR, R0                          ; 0546
                                  63  11 00091            BRB      14$                                         ;
                    02            63  B1 00093  8$:       CMPW     (R3), #2                                    ; 0551
                                  1C  12 00096            BNEQ     10$                                         ;
                    50      2C    A7  D0 00098            MOVL     TAPE_XFER_ADDR, R0                          ; 0553
                    52            64  3C 0009C            MOVZWL   (R4), R2                                    ;
                                6042  D5 0009F            TSTL     (R0)[R2]                                    ;
                                  0A  12 000A2            BNEQ     9$                                          ;
                    50      24    A7  D0 000A4            MOVL     TAPE_IMAGE, R0                              ; 0554
                                6042  7F 000A8            PUSHAQ   (R0)[R2]                                    ;
                    68            01  FB 000AB            CALLS    #1, GET_IMAGE                               ;
                    50      2C    A7  D0 000AE  9$:       MOVL     TAPE_XFER_ADDR, R0                          ; 0555
                                  65  11 000B2            BRB      17$                                         ;
                    20            63  B1 000B4  10$:      CMPW     (R3), #32                                   ; 0560
                                  1C  12 000B7            BNEQ     12$                                         ;
                    50      20    A7  D0 000B9            MOVL     SCOM_XFER_ADDR, R0                          ; 0562
                    52            64  3C 000BD            MOVZWL   (R4), R2                                    ;
                                6042  D5 000C0            TSTL     (R0)[R2]                                    ;
                                  0A  12 000C3            BNEQ     11$                                         ;
                    50      18    A7  D0 000C5            MOVL     SCOM_IMAGE, R0                              ; 0563
                                6042  7F 000C9            PUSHAQ   (R0)[R2]                                    ;
                    68            01  FB 000CC            CALLS    #1, GET_IMAGE                               ;
                    50      20    A7  D0 000CF  11$:      MOVL     SCOM_XFER_ADDR, R0                          ; 0564
                                  67  11 000D3            BRB      20$                                         ;
           0080     8F            63  B1 000D5  12$:      CMPW     (R3), #128                                  ; 0569
                                  1C  12 000DA            BNEQ     15$                                         ;
                    50      E0    A7  D0 000DC            MOVL     BUS_XFER_ADDR, R0                           ; 0571
                    52            64  3C 000E0            MOVZWL   (R4), R2                                    ;
                                6042  D5 000E3            TSTL     (R0)[R2]                                    ;
                                  0A  12 000E6            BNEQ     13$                                         ;
                    50      D8    A7  D0 000E8            MOVL     BUS_IMAGE, R0                               ; 0572
                                6042  7F 000EC            PUSHAQ   (R0)[R2]                                    ;
                    68            01  FB 000EF            CALLS    #1, GET_IMAGE                               ;
                    50      E0    A7  D0 000F2  13$:      MOVL     BUS_XFER_ADDR, R0                           ; 0573
                                  44  11 000F6  14$:      BRB      20$                                         ;
           0060     8F            63  B1 000F8  15$:      CMPW     (R3), #96                                   ; 0577
                                  1C  12 000FD            BNEQ     18$                                         ;
                    50      14    A7  D0 000FF            MOVL     REALTIME_XFER_ADDR, R0                      ; 0579
                    52            64  3C 00103            MOVZWL   (R4), R2                                    ;
                                6042  D5 00106            TSTL     (R0)[R2]                                    ;
                                  0A  12 00109            BNEQ     16$                                         ;
                    50      0C    A7  D0 0010B            MOVL     REALTIME_IMAGE, R0                          ; 0580
                                6042  7F 0010F            PUSHAQ   (R0)[R2]                                    ;
                    68            01  FB 00112            CALLS    #1, GET_IMAGE                               ;
                    50      14    A7  D0 00115  16$:      MOVL     REALTIME_XFER_ADDR, R0                      ; 0581
                                  21  11 00119  17$:      BRB      20$                                         ;
           0046     8F            63  B1 0011B  18$:      CMPW     (R3), #70                                   ; 0584
                                  20  12 00120            BNEQ     21$                                         ;
                    50      3C    A7  D0 00122            MOVL     WORKSTATION_XFER_ADDR, R0                   ; 0586
                    52            64  3C 00126            MOVZWL   (R4), R2                                    ;
                                6042  D5 00129            TSTL     (R0)[R2]                                    ;
                                  0A  12 0012C            BNEQ     19$                                         ;
                    50      34    A7  D0 0012E            MOVL     WORKSTATION_IMAGE, R0                       ; 0587
                                6042  7F 00132            PUSHAQ   (R0)[R2]                                    ;
                    68            01  FB 00135        CALLS   #1, GET_IMAGE                         ;
                    50        3C  A7  D0 00138 19$:   MOVL    WORKSTATION_XFER_ADDR, R0             ; 0588
                    65          6042  D0 0013C 20$:   MOVL    (R0)[R2], (R5)                        ;
                                  0E  11 00140        BRB     22$                                   ; 0512
                    7E            63  3C 00142 21$:   MOVZWL  (R3), -(SP)                           ; 0596
                                  01  DD 00145        PUSHL   #1                                    ;
                       00000000G  8F  DD 00147        PUSHL   #ERF_UNKCLASS                         ;
                    69            03  FB 0014D        CALLS   #3, LIB$SIGNAL                        ;
                    50            01  D0 00150 22$:   MOVL    #1, R0                                ; 0599
                                      04 00153        RET                                           ; 0600

; Routine Size:  340 bytes,    Routine Base:  $CODE + 00BD
;  320        0601  1 Routine GET_IMAGE (Image_name) =
;  321        0602  2 Begin
;  322        0603  2 !++
;  323        0604  2 ! Functional description
;  324        0605  2 !
;  325        0606  2 !       This routine takes the image name specified, test to see that it
;  326        0607  2 !       is not of zero length and then calls map_image to load the image.
;  327        0608  2 !       If the image gets loaded, it is called in order to obtain the
;  328        0609  2 !       the transfer vector information.
;  329        0610  2 !
;  330        0611  2 ! Calling sequence
;  331        0612  2 !
;  332        0613  2 !       Get_image (Image_name)
;  333        0614  2 !
;  334        0615  2 ! Input parameters
;  335        0616  2 !
;  336        0617  2 !       Image_name : Contains address of image_name descriptor
;  337        0618  2 !
;  338        0619  2 ! Output parameters
;  339        0620  2 !
;  340        0621  2 !       None
;  341        0622  2 !
;  342        0623  2 ! Routine value
;  343        0624  2 !
;  344        0625  2 !       Worst error is returned.
;  345        0626  2 !
;  346        0627  2 !----
;  347        0628  2
;  348        0629  2
;  349        0630  2 Local
;  350        0631  2         Array_addr,
;  351        0632  2         Array_size,
;  352        0633  2         Class:                  WORD SIGNED,
;  353        0634  2         Status,
;  354        0635  2         Temp_xfer_addr,
;  355        0636  2         Transfer_offset:        WORD,
;  356        0637  2         Type:                   WORD,
;  357        0638  2         Version:                WORD,
;  358        0639  2         Xfer_addr;
;  359        0640  2
;  360        0641  2 Bind
;  361        0642  2    Desc = .Image_name  : $BBLOCK[dsc$k_d_bln] ;
;  362        0643  2
;  363        0644  2 Literal
;  364        0645  2    Tran_vec_size = 8;
;  365        0646  2
```

```
2 !
2 ! If the Image name is not blank then try to load the image.
2 !
2
2 If .desc[dsc$w_length] EQL 0 then (Xfer_addr = 0; Return true);
2
2 Status = map_image ( desc, Xfer_addr ) ;
2 If NOT .status then (Xfer_addr = 0; Return true);
2
2
2 !       Call the image that was just loaded and recieve from it
2 !       the address and size of an array of longword codes. Each
2 !       of these codes are of the form:
2 !       31                             0
2 !       -------------------------------
2 !       ! Device_class  ! Device_type !
2 !       -------------------------------
2 !       !Transfer_offset!  Version    !
2 !       -------------------------------
2 !
2 !       The first longword contains the class and type, while the
2 !       second longword contains the version number of each
2 !       device module in the loaded image, along with the transfer
2 !       vector offset value.
2 !
2
2 Exec_image ( Xfer_addr, Array_addr, Array_size);
2
2 Incr I from 1 to .Array_size do
3   Begin
3   Get_code ( Array_addr, Class, Type, Version, Transfer_offset );
3
3   Temp_xfer_addr = .xfer_addr + tran_vec_size * .transfer_offset;
3
3   Selectoneu .Class of set
3
3   [DC$_ZERO_CLASS]:
4         Begin
4           Incr J from 1 to .max_misc_type do
5             Begin
5             If .Translate_entry_table [.J] EQL .Type then
6               Begin
6               If .Packet_processor_version [.J] EQL .version then
6                   Packet_processor_xfer_addr [.J] = .temp_xfer_addr
6               Else
6                   Packet_processor_xfer_addr [.J] = 0;
6               Class = -1;
6               Exitloop;
5                 End;
4             End;
4
4             If .class LSS 0 then
4               class = 0
4               Else
5               Begin
5               Signal (erf_unkentry, 1, .type);
5               Class = 0;
```

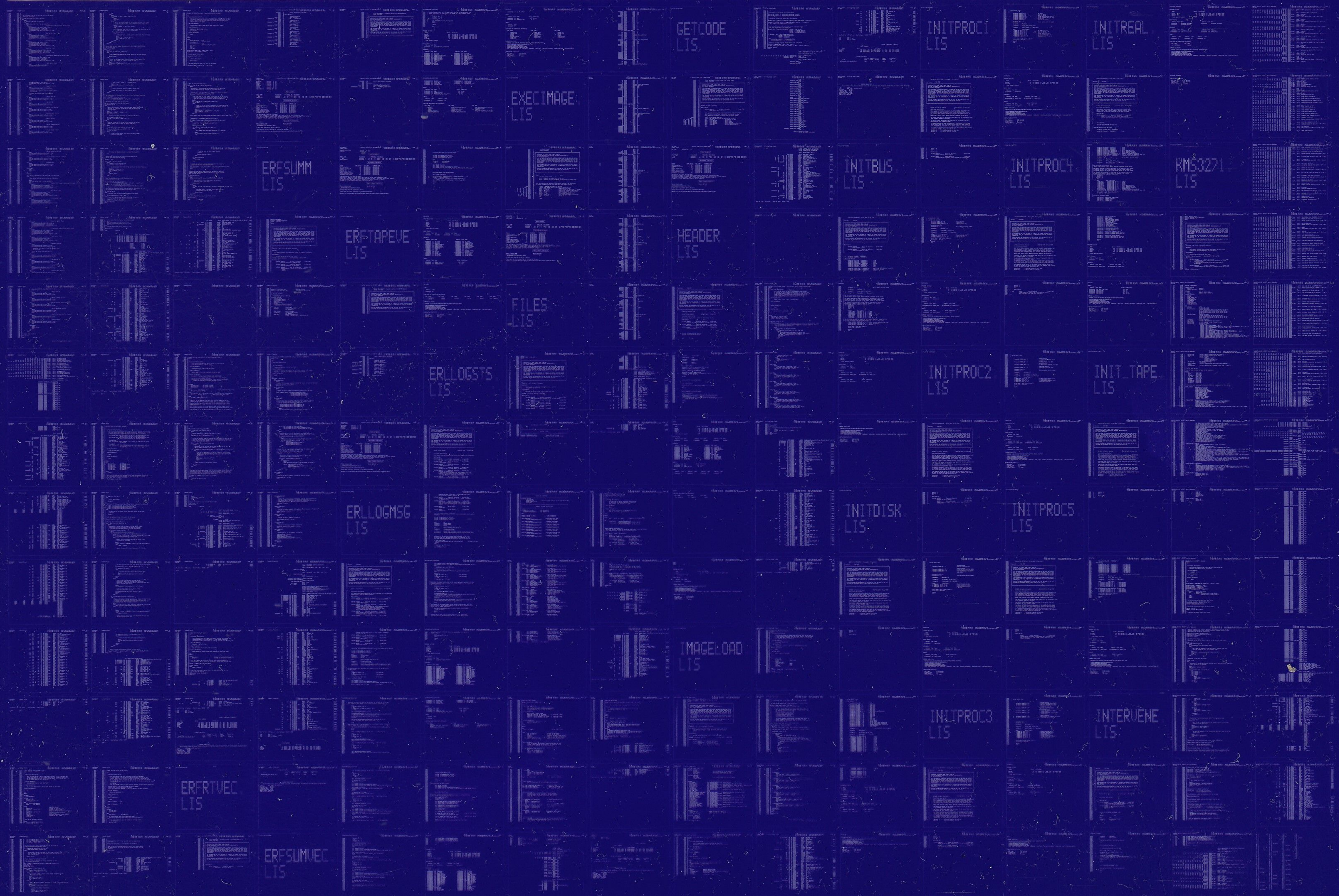
```
;  424         0704  4                   End;
;  425         0705  3               End;
;  426         0706  3
;  427         0707  3
;  428         0708  3       [DC$_DISK]:
;  429         0709  4               Begin
;  430         0710  4               Disk_xfer_addr [.type] = .temp_xfer_addr;
;  431         0711  4               Disk_version [.type] = .version;
;  432         0712  3               End;
;  433         0713  3
;  434         0714  3
;  435         0715  3       [DC$_TAPE]:
;  436         0716  4               Begin
;  437         0717  4               Tape_xfer_addr [.type] = .temp_xfer_addr;
;  438         0718  4               Tape_version [.type] = .version;
;  439         0719  3               End;
;  440         0720  3
;  441         0721  3
;  442         0722  3       [DC$_BUS]:
;  443         0723  4               Begin
;  444         0724  4               Bus_xfer_addr [.type] = .temp_xfer_addr;
;  445         0725  4               Bus_version [.type] = .version;
;  446         0726  3               End;
;  447         0727  3
;  448         0728  3
;  449         0729  3       [DC$_SCOM]:
;  450         0730  4               Begin
;  451         0731  4               Scom_xfer_addr [.type] = .temp_xfer_addr;
;  452         0732  4               Scom_version [.type] = .version;
;  453         0733  3               End;
;  454         0734  3
;  455         0735  3       [DC$_REALTIME]:
;  456         0736  4               Begin
;  457         0737  4               Realtime_xfer_addr [.type] = .temp_xfer_addr;
;  458         0738  4               Realtime_version [.type] = .version;
;  459         0739  3               End;
;  460         0740  3
;  461         0741  3       [DC$_WORKSTATION]:
;  462         0742  4               Begin
;  463         0743  4               Workstation_xfer_addr [.type] = .temp_xfer_addr;
;  464         0744  4               Workstation_version [.type] = .version;
;  465         0745  3               End;
;  466         0746  3
;  467         0747  3       [OTHERWISE]:
;  468         0748  3   !signal maybe
;  469         0749  3       TES;
;  470         0750  2   End;
;  471         0751  2
;  472         0752  2   Return true;
;  473         0753  1   End;




                                              007C 00000 GET_IMAGE:
                                                         .WORD   Save R2,R3,R4,R5,R6                          ; 0601
```

```
                     56 00000000'    00  9E 00002          MOVAB   MAX_MISC_TYPE, R6                     ;
                     5E              1C  C2 00009          SUBL2   #28, SP                               ;
                              04     BC  B5 0000C          TSTW    @IMAGE_NAME                           ; 0651
                                     0E  13 0000F          BEQL    1$                                    ;
                              04     AE  9F 00011          PUSHAB  XFER_ADDR                             ; 0653
                              04     AC  DD 00014          PUSHL   IMAGE_NAME                            ;
              FDD3   CF              02  FB 00017          CALLS   #2, MAP_IMAGE                         ;
                     06              50  E8 0001C          BLBS    STATUS, 2$                            ; 0654
                              04     AE  D4 0001F  1$:     CLRL    XFER_ADDR                             ;
                                   00EE  31 00022          BRW     20$                                   ;
                                     5E  DD 00025  2$:     PUSHL   SP                                    ; 0673
                              1C     AE  9F 00027          PUSHAB  ARRAY_ADDR                            ;
                              0C     AE  9F 0002A          PUSHAB  XFER_ADDR                             ;
         00000000G   00              03  FB 0002D          CALLS   #3, EXEC_IMAGE                        ;
                                     55  D4 00034          CLRL    I                                     ; 0679
                                     74  11 00036          BRB     10$                                   ;
                              08     AE  9F 00038  3$:     PUSHAB  TRANSFER_OFFSET                       ; 0677
                              10     AE  9F 0003B          PUSHAB  VERSION                               ;
                              18     AE  9F 0003E          PUSHAB  TYPE                                  ;
                              20     AE  9F 00041          PUSHAB  CLASS                                 ;
                              28     AE  9F 00044          PUSHAB  ARRAY_ADDR                            ;
         00000000G   00              05  FB 00047          CALLS   #5, GET_CODE                          ;
                     50       08     AE  3C 0004E          MOVZWL  TRANSFER_OFFSET, R0                   ; 0679
                     53       04   BE40  7E 00052          MOVAQ   @XFER_ADDR[R0], TEMP_XFER_ADDR        ;
                     51       14     AE  32 00057          CVTWL   CLASS, R1                             ; 0681
                                     51  12 0005B          BNEQ    11$                                   ; 0683
                     54              66  9A 0005D          MOVZBL  MAX_MISC_TYPE, R4                     ; 0685
                                     50  D4 00060          CLRL    J                                     ; 0687
                                     29  11 00062          BRB     7$                                    ;
                     51       34     A6  D0 00064  4$:     MOVL    TRANSLATE_ENTRY_TABLE, R1             ;
                10   AE            6140  B1 00068          CMPW    (R1)[J], TYPE                         ;
                                     1E  12 0006D          BNEQ    7$                                    ;
                     51       04     A6  D0 0006F          MOVL    PACKET_PROCESSOR_XFER_ADDR, R1        ; 0690
                     52       0C     A6  D0 00073          MOVL    PACKET_PROCESSOR_VERSION, R2          ; 0689
                0C   AE            6240  B1 00077          CMPW    (R2)[J], VERSION                      ;
                                     06  12 0007C          BNEQ    5$                                    ;
                   6140              53  D0 0007E          MOVL    TEMP_XFER_ADDR, (R1)[J]               ; 0690
                                     03  11 00082          BRB     6$                                    ;
                                   6140  D4 00084  5$:     CLRL    (R1)[J]                               ; 0692
                14   AE              01  AE 00087  6$:     MNEGW   #1, CLASS                             ; 0693
                                     04  11 0008B  8$:     BRB     8$                                    ; 0688
        D3           50              54  F3 0008D  7$:     AOBLEQ  R4, J, 4$                             ; 0685
                              14     AE  B5 00091  8$:     TSTW    CLASS                                 ; 0698
                                     13  19 00094          BLSS    9$                                    ;
                     7E       10     AE  3C 00096          MOVZWL  TYPE, -(SP)                           ; 0702
                                     01  DD 0009A          PUSHL   #1                                    ;
                       00000000G     8F  DD 0009C          PUSHL   #ERF_UNKENTRY                         ;
         00000000G   00              03  FB 000A2          CALLS   #3, [IB$SIGNAL                        ;
                              14     AE  B4 000A9  9$:     CLRW    CLASS                                 ; 0703
                                     5F  11 000AC  10$:    BRB     19$                                   ; 0681
                     01              51  B1 000AE  11$:    CMPW    R1, #1                                ; 0708
                                     0E  12 000B1          BNEQ    13$                                   ;
                     51       EC     A6  7D 000B3          MOVQ    DISK_VERSION, R1                      ; 0711
                     50       10     AE  3C 000B7  12$:    MOVZWL  TYPE, R0                              ; 0710
                   6240              53  D0 000BB          MOVL    TEMP_XFER_ADDR, (R2)[R0]              ;
                                     47  11 000BF          BRB     18$                                   ; 0711
                     02              51  B1 000C1  13$:    CMPW    R1, #2                                ; 0715
```

```
                                                06 12 000C4          BNEQ    14$                           ;
                                   51      2C   A6 7D 000C6          MOVQ    TAPE_VERSION, R1              ; 0718
                                                EB 11 000CA          BRB     12$                           ; 0717
                           0080    8F           51 B1 000CC 14$:     CMPW    R1, #128                      ; 0722
                                                06 12 000D1          BNEQ    15$                           ;
                                   51      E0   A6 7D 000D3          MOVQ    BUS_VERSION, R1               ; 0725
                                                DE 11 000D7          BRB     12$                           ; 0724
                                   20           51 B1 000D9 15$:     CMPW    R1, #32                       ; 0729
                                                06 12 000DC          BNEQ    16$                           ;
                                   51      20   A6 7D 000DE          MOVQ    SCOM_VERSION, R1              ; 0732
                                                D3 11 000E2          BRB     12$                           ; 0731
                           0060    8F           51 B1 000E4 16$:     CMPW    R1, #96                       ; 0735
                                                06 12 000E9          BNEQ    17$                           ;
                                   51      14   A6 7D 000EB          MOVQ    REALTIME_VERSION, R1          ; 0738
                                                C6 11 000EF          BRB     12$                           ; 0737
                           0046    8F           51 B1 000F1 17$:     CMPW    R1, #70                       ; 0741
                                                15 12 000F6          BNEQ    19$                           ;
                                   51      40   A6 D0 000F8          MOVL    WORKSTATION_XFER_ADDR, R1     ; 0743
                                   50      10   AE 3C 000FC          MOVZWL  TYPE, R0                      ;
                                 6140           53 D0 00100          MOVL    TEMP_XFER_ADDR, (R1)[R0]      ;
                                   51      3C   A6 D0 00104          MOVL    WORKSTATION_VERSION, R1       ; 0744
                                 6140      0C   AE B0 00108 18$:     MOVW    VERSION, (R1)[R0]             ;
       FF25          55            01           6E F1 0010D 19$:     ACBL    ARRAY_SIZE, #1, I, 3$         ; 0675
                                   50           01 D0 00113 20$:     MOVL    #1, R0                        ; 0752
                                                   04 00116          RET                                   ; 0753

; Routine Size:  279 bytes,    Routine Base:  $CODE + 0211


;  474          0754  1
;  475          0755  1 END
;  476          0756  0 ELUDOM




                                                                     .EXTRN  LIB$SIGNAL, LIB$STOP

;                                 PSECT SUMMARY
;
;       Name                  Bytes                    Attributes
;
;   $GLOBAL$                    108  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;   $PLIT                        20  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;   $CODE                       808  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)




;                            Library Statistics
;
;                                      -------- Symbols --------       Pages       Processing
;       File                           Total    Loaded   Percent       Mapped      Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1       18619        45         0       1000          00:02.0
;                                               COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:IMAGELOAD/OBJ=OBJ$:IMAGELOAD MSRC$:IMAGELOAD/UPDATE=(ENH$:IMAGELOAD)

;  477           0757  0
; Size:          808 code + 128 data bytes
; Run Time:         00:19.5
; Elapsed Time:     00:40.7
; Lines/CPU Min:     2332
; Lexemes/CPU-Min: 20268
; Memory Used:  180 pages
; Compilation Complete
```

0149 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



GETCODE
LIS

INITPROC1
LIS

INITREAL
LIS

EXECIMAGE
LIS

ERFSUMM
LIS

INITBUS
LIS

INITPROC4
LIS

KMS3271
LIS

ERFTAPEVE
LIS

HEADER
LIS

FILES
LIS

ERLLOGSTS
LIS

INITPROC2
LIS

INIT_TAPE
LIS

ERLLOGMSG
LIS

INITDISK
LIS

INITPROC5
LIS

IMAGELOAD
LIS

INITPROC3
LIS

INTERVENE
LIS

ERFRTVEC
LIS

ERFSUMVEC
LIS